CS6362 - Project Report

Sanidhya Mangal
Vanderbilt University
sanidhya.v.mangal@vanderbilt.edu

1 Introduction

The aim of this project is to implement Mishra et al. [2021] for testing out the certain hypothesis and find limitations of the approach if any. Mishra et al. [2021] try to solve the problem of domain adaptation by proposing a semi-supervised method, i.e., a paradigm in machine learning where we try to learn the model well on one or more source distribution and then try to use this trained model on a target data or distribution and in semi-supervised setting small subset of data from target domain is included. More details about this could be found in later sections.

Upon reading and analyzing the article some hypothesis were drawn which are as follows:

- In the article author has mentioned about the pre-training the model using rotations, as proposed by Gidaris et al. [2018]. And they have only compared this method with baseline pretained imagenet Deng et al. [2009] weights, but there are many other unsupervised visual pre-training methods leveraging contrastive methods. Hence, one of the hypothesis proposed is to juxtapose the proposed method on several different pre-train activation's and record the model performance on the same. Therefore the experiment was run on the ImageNet, RotationNet and Supervised Contrastive learning Khosla et al. [2020] based pre training.
- In semi-supervised domain adaptation, a very small subset of target data is included along with the source distribution while training the models, but generally all the classes or subclasses from target domain are included in the training of the models. However, there is a small variation in this i.e., only a subset of classes or subclasses from target models are included. Lets say there are N classes in all the domains but while training only M classes are picked randomly from the target domain instead all the classes. Based on this variation another hypothesis was drawn to test the proposed algorithm on this variation.
- Authors claims that their method performs well in the semi-supervised setting, but there was no evidence if this method could work in unsupervised setting or not. Therefore, one of the hypothesis proposed is to test what happens in case of multi-domain case. i.e., lets say there are N domains then we will treat N-i^{th} domains as source domains and then test the model performance on that i^{th} domain.

To validate all the hypothesis all the pretraining was used and the results are reported in Section 4.3. The report is organized in following manner, in later section we compare the related work followed by method in Section 3, experimental details such as dataset, implementation and results in 4 and in the end discussion and conclusion about the experiments is provided.

2 Related Work

Firstly, Saito et al. [2019] is explored to understand how domain adaptation works in semi-supervised setting, in their work they mainly talk about Minmax entropy (MME). In this setting to improve the performance on in target domain we try to maximize the entropy on classifier nodes and minimize it in feature extraction in an attempt to move the boundary close to and in between target unlabelled examples, it is one of the baseline work and Mishra et al. [2021] draws a comparison and performance

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

of PAC with respect to MME. Since one of the key analogy we aim to draw is by testing various pre-trained models, hence to understand more about such methods Gidaris et al. [2018], is explored to learn how unsupervised visual activation using works. To learn more about contrastive pre-training works we need to extensively study He et al. [2020],Oord et al. [2018] and use one of the method to pre-train the feature extractor backbone. In addition to this other pre-training methods such as SimSiam could also be embodied Chen and He [2021]. As it can be seen that Mishra et al. [2021] focuses more on semi-supervised learning but recently Sun et al. [2019] has shown that if we perform auxilary self-supervised learning simultaneously in target domain and source domain then it can generalize well on the unlabelled data and brings both the domain closer together along the direction of relevant tasks, i.e., it renders a way to draw unsupervised domain adaptation from semi-supervised method . As it is evident from the paper that we need both normal images and perturbed images for domain adaptation task therefore we need to explore Cubuk et al. [2020] for learning how to apply random transformation on the images to generate perturbed images along with the random color jitters to feed into models for computing consistency. Sajjadi et al. [2016] could also be explored to study more about the stochastic transformation and perturbations for deep semi-supervised domain adaptation. Sohn et al. [2020],Shu et al. [2018],Verma et al. [2019],Xu et al. [2019] also explores Semi-Supervised domain adaptation in various angles such as interpolation and consistency on large scale computer vision task. Lastly, Wallace and Hariharan [2020] is also a good read to understand how to extend semi-supervised learning for domain adaptation.

3 Method

The entire training is a two step process, therefore first we need to learn the representations using some unsupervised and supervised method before fine tuning the image classifier for the domain adaptation task. In this experiment three different strategies are used, one is the normal vanilla method that is pre training the network as a classifier on the imagenet Deng et al. [2009] dataset. Second one is unsupervised approach for learning rotations as proposed by Gidaris et al. [2018], they proposed a method to learn the feature representations using Convolution nets to recognize the 2d rotations applied on the images. Four different rotations of 90,180,270,360 degree are applied on all the images from all the domains and then these rotations are fed into the neural network which follows the self-supervisory signal to learn the features. Fig. 1 illustrates the flow of information and learning strategy used in rotation based unsupervised feature learning.

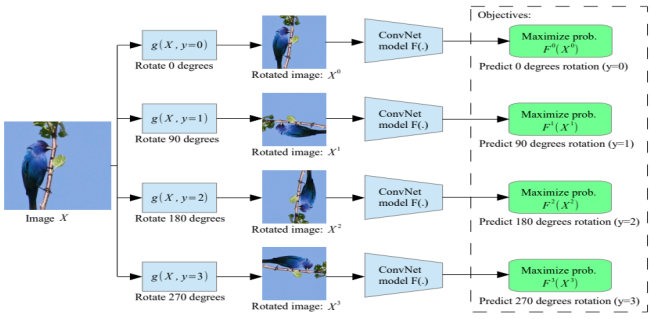

Figure 1: Illustration of self supervisory task proposed by Gidaris et al. [2018] for feature learning using rotational transformations

The third feature representation learning is carried using supervised contrastive learning Khosla et al. [2020], a self-supervised setting is used in this experiment. It contains three components which are

- *Augmentor*, used to transform the images by applying color jitter, color drop, etc. but in this experiment only color drop and color jitter is used.
- *Encoder network*, it maps the augmented images $\hat{x}$ to a representation vector $r = E(\hat{x})$, In the proposed method the final activations from the pooling layer are used as representation vector which are normalized to the unit vector sphere.

- *projector network*, it maps the representational vector r to the vector $z = P(r)$ which is suitable for the contrastive loss. For this experiment we use a multi layer perceptron (MLP) suitable for contrastive loss. The vector z is again normalized on the unit vector sphere.

To train the network on self-supervised contrastive representation we use loss defined in eq. 1 and eq. 2.

$$\mathcal{L}^{\text{self}} = \sum_{i=1}^{2N} \mathcal{L}_i^{\text{self}} \tag{1}$$

$$\mathcal{L}_i^{\text{self}} = -\log \frac{\exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_{j(i)}/\tau\right)}{\sum_{k=1}^{2N} \mathbb{1}_{i \neq k} \cdot \exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_k/\tau\right)} \tag{2}$$

Where, $z = P(E(\hat{x}))$ and 1 is an indicator function.

Once, the model has learned various representations we can commence with the second step in the training i.e., fine tuning our image classifier model. This model can be decomposed into two different components, which are feature extractor and classifier. Feature extractor F is the generally a ConvNet based backbone, in this experiment resnet50 is used as the feature extractor. Next it classifier C, it is a MLP with out being the probability distribution of the classes. The final output of the model could be obtained by $C(F(x)) \in R^K$, where K denotes the number of categories an image could belong to.

The approach described in the paper relies heavily on pretraining and consistency (PAC). Before describing the approach or the method used in the paper. I'd like to introduce some notations beforehand, D_s refers to the dataset from source domain, D_t refers to the small set of data from the target domain distribution. Both the data distribution are supervised i.e., all the data points are of form $\{x_i, y_i\}$. In addition to this, one more distribution is taken into account i.e., D_u which consists of unlabeled data points from the target domain.

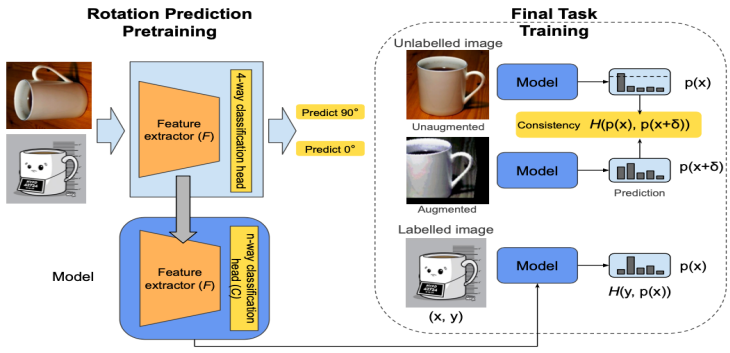

Figure 2: A diagram of PAC approach.

In the paper they defined rotation pretraining as there primary pre training strategy but in this experiment supervised contrastive learning and vanilla method are also used, as explained earlier. Now in order to train or prepare our model for domain adaptation task we need to perform the consistency regularization on our final model $F \circ C$ to produce same results for our normal images and perturbed images denoted by x and $x + \delta$ respectively. Given the $x \in D_u$ we first compute the models prediction in for both the setting as described in eq. 3

$$\begin{aligned} p_x &= p(y \mid x; F, C) = \text{softmax}(C(F(x))) \\ q_x &= p(y \mid x + \delta; F, C) = \text{softmax}(C(F(x + \delta))) \end{aligned} \tag{3}$$

p_x is then thresholded with τ to compute consistency loss described in eq. 4

$$\mathcal{L}_{CR}(x) = \mathbb{1}\left[\max_{k \in K} p_x(k) \geq \tau\right] H\left(p_x, q_x\right) \tag{4}$$

where $H(p_x, q_x)$ is cross entropy loss defined by $H\left(p_x, q_x\right) = \sum_{k \in K} -p_x(k) \log\left(q_x(k)\right)$ and 1 is indicator function.

One catch here is the computation of gradients, to compute gradients of network p_x is not used and we use ground truth as a target. The model is trained using mini batch SDG with minibatch M_s, M_t and M_u drawn over distribution D_s, D_t and D_u respectively. The mini batch loss is given by:

$$\mathcal{L} = \frac{1}{|M_s|} \sum_{(\boldsymbol{x}, y) \in M_s} H(\bar{y}, \boldsymbol{x}) + \frac{1}{|M_t|} \sum_{(\boldsymbol{x}, y) \in M_t} H(\bar{y}, \boldsymbol{x}) + \frac{1}{|M_u|} \sum_{\boldsymbol{x} \in M_u} \mathcal{L}_{CR}(\boldsymbol{x}) \tag{5}$$

Where, $\bar{y} \in R^K$ is one hot representation in $y \in [K]$. More details about the PAC could be found in Fig. 2

4 Experiment

4.1 Dataset

All the hypothesis are tested on Office-Home Venkateswara et al. [2017] dataset. It is a small scale dataset to evaluate domain adaptation algorithms using deep learning. It contains images from multiple domains which are Artistic, Clipart, Real World and Product. In all the domains the dataset contains everyday objects modeled into 65 categories, eg. cup, pen, table, clock, hammer, etc. It can be observed from the Fig. 3 that **Artistic** domain contains images mostly from the painting, sketches, etc. **Clipart** domain: clipart images, **Real World**: images captured via mobile phone cameras and **Product**: images with proper lighting and background. All the images were reshaped to 128x128 pixel size for the entire experiment. In addition to this, around 10% of data was used as a target dataset from the target domain and around 3 images were left from each class for the validation set.

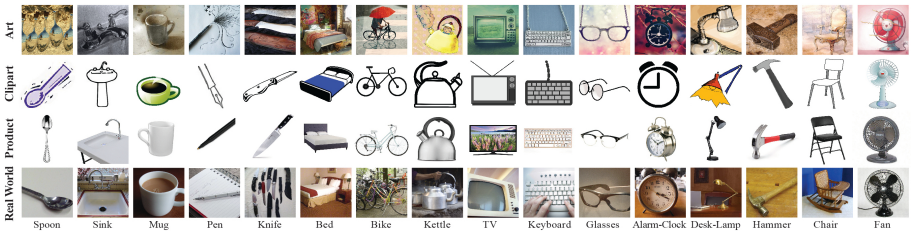

Figure 3: Sample image containing Office Home dataset.

4.2 Implementation

All the implementation was done using Tensorflow and Keras Abadi et al. [2016]. Pillow Clark [2015] was used to preprocess the rotation dataset for rotation training. To test out the hypothesis around the representation learning, we first need to train Resnet50He et al. [2016] model for unsupervised rotation learning Gidaris et al. [2018] and supervised contrastive learning Khosla et al. [2020]. After successfully training the model on these representation the weights of base model i.e., resnet50 are saved to be used later. Since the entire training is a two step process hence to formulate image classifier model we need to design a model. This model has two component one is feature extractor and another one is classifier. Resnet50 is used as a feature extractor and for classifier model a two layered multi layered perceptron is used with 512 units in each layer, to prevent over-fitting in the model dropout rate of 0.3 was set. The final output from the image classifier model was a softmax probability distribution of the shape of the class. SGD optimizer was used for backprop with initial learning rate of 0.001 and momentum of 0.9. To sharpen the training cosine annealing Loshchilov and Hutter [2016] was used with the decay steps as 1000. For consistency regularization the confidence threshold τ was set to 0.9 was used through out the experiments.

In addition to the models, for consistency regularization and supervised contrastive learning transformation of data is required too. Hence, to carry this out image ops from Tensorflow is used to apply random color jitter, hue, flip, etc. This transformation module is coupled with the data loader pipeline to load the datasets, preprocess them and then feed them to the models.

All the experiments were conducted for all three representation learning approaches. For the domain adaptation following pair were used, Product $\rightarrow$ Real World and Real World $\rightarrow$ Art. The entire hyperparamerters were kept same for all the experiments. In addition to this, to record the progress of the experiments a csv log file was generate to keep the track of progress through out the experiments. Due to limitation of compute resources all the experiments were executed for 50 epochs only. After successfully training the models, all the models were evaluated on validation after completion of training process. Further details about all the results could be found in 4.3.

4.3 Results

The loss and accuracy of each training steps for the Rotnet pretraining could be visualized in Fig. 4 and 5. The blue line describes the loss and accuracy on the validation set whereas the orange line describes the accuracy and loss for the training dataset.

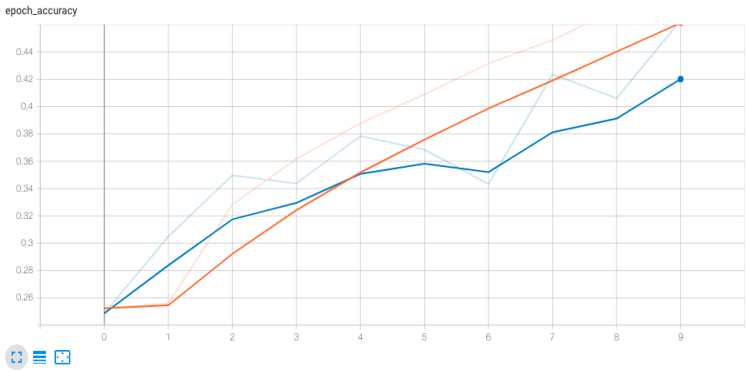

Figure 4: Rotation Pretraining Accuracy

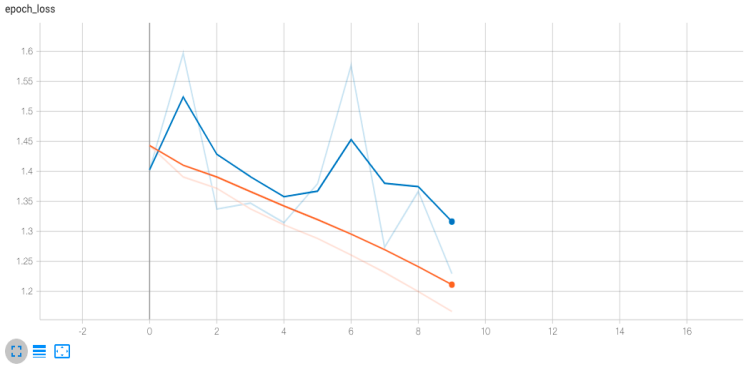

Figure 5: Rotation Pretraining Loss

In addition to this the pretraining of supervised contrastive learning could be visualized in Fig. 6.

The performance of all the experiments could be visualized in Table 1, Table 2 and Table 3. All the tables describes the accuracy score in range of 0 to 1. Closer score is to 1 better the model is.

In the tables we record 3 parameter which are, Weights defining the pre training weights used in the experiments it could be the choice between Imagenet, RotNet and SupCon where rotation net defines the pretraining using Gidaris et al. [2018] and SupCon defines Supervised Contrastive learning based pre training Khosla et al. [2020]. Along with the weights there is another column which indicates the

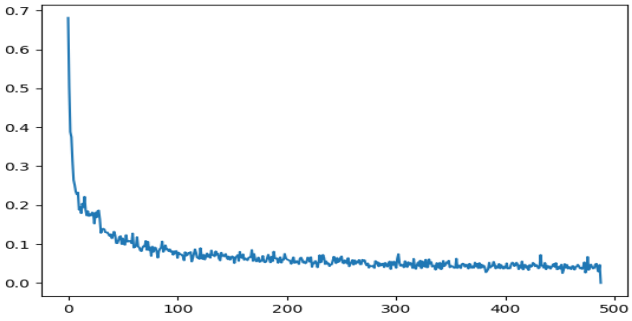

Figure 6: Supcon Pretraining Loss

source domain and target domain. Lastly, there is a column storing accuracy score for that particular experiment. In all the settings or the hypothesis the layout for the table is consistent.

Weights	Source → Target	Accuracy Score
ImageNet	Product → Real World	0.55
ImageNet	Real World → Art	0.24
Rotnet	Product → Real World	0.13
Rotnet	Real World → Art	0.09
SupCon	Product → Real World	0.11
SupCon	Real World → Art	0.08

Table 1: Semi Supervised Accuracy Metrics

Weights	Source → Target	Accuracy Score
ImageNet	Product → Real World	0.53
ImageNet	Real World → Art	0.22
Rotnet	Product → Real World	0.11
Rotnet	Real World → Art	0.05
SupCon	Product → Real World	0.07
SupCon	Real World → Art	0.12

Table 2: Semi Supervised Variant Accuracy Metrics

It could be clearly visualized from Fig. 7, 8 and 9 that the loss is decreasing as training steps increases. In these figures the loss for the semi supervised setting and variation in the semi supervised setting is presented to render that the models are converging despite of this behavior the performance obtained is not much.

5 Discussion

Despite of following the exact experimentation details as described in the paper Mishra et al. [2021] it was hard to obtain the similar performance as described in the paper the top accuracy was obtained on the Imagenet pretraining which shows an aberration to the results documented. It could be noted that the highest accuracy achieved was of 55% under Product to Real World domain shift but in case of other pretraining it failed to produce the better results. Though only better than the chance accuracy i.e., $1/K$, where K defines the total number of categories. This is little strange as the performance of the pretraining was pretty good and as seen in the pretraining logs both rotation strategy and supcon strategy projected high confidence results and in case of rotation strategy they were pretty comparable to the results quoted in the article. One reason I could think for such performance could be the selection of feature extractor more. I believe that the model selected was very dense as per the article and in my notion if we replace ResNet with a simpler model such as Resnet18 or AlexNet34 it might render better results since there might be a chance that the feature extractor model is overfitting

Weights	Source → Target	Accuracy Score
ImageNet	Product → Real World	0.006
ImageNet	Real World → Art	0.07
Rotnet	Product → Real World	0.10
Rotnet	Real World → Art	0.10
SupCon	Product → Real World	0.07
SupCon	Real World → Art	0.05

Table 3: Un Supervised Accuracy Metrics

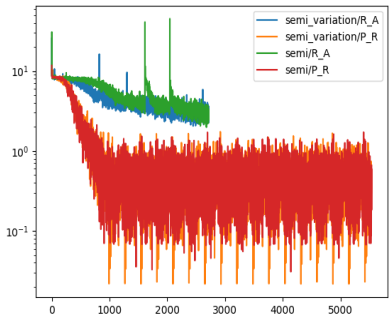

Figure 7: Imagenet Loss

due to the depth. One such reason to bolster my claim is the pre training of the model during rotation strategy as it was quoted in the paper that they trained the model for only 5000 iterations and were able to achieve the accuracy of around 53% but in the case of ResNet50 it took around 10000 training steps to achieve a result of around 40% on the validation set which is still lower than the accuracy quoted in the paper. Hence, it might be due to the denser model.

In addition to this, based on the validation results it was noticed that the there was very significant loss in the accuracy of the models when some random classes from the labeled target distribution was dropped the change of accuracy was approx -2% in all the domain shifts. In addition to this, it was anticipated that there would be a huge drop in the performance for the unsupervised setting but there was hardly any difference in that too. It could be seen that the results are somewhat close to the ones with semi supervised. However, there is a huge drop in the performance of Imagenet for the unsupervised setting.

Besides, this one more notable conclusion could be inferred from the results obtained that the change in source and target domain yields different performance which tends to validate with the other domain adaptation paper which could be treated as a good sign. In this scenario, Product → Real

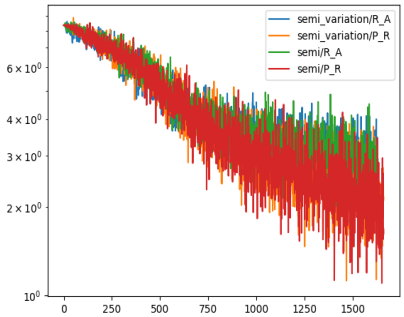

Figure 8: RotNet Loss

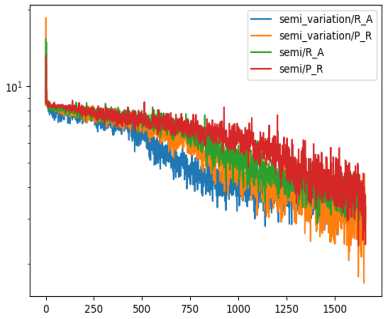

Figure 9: SupCon Loss

World tends to outperform Real World → Art. However, in semi supervised variant scenarios SupCon model tends to outperform the Product to Real World by around 5%.

One of the notable conclusions which could be while looking the loss curves is that despite of drop in loss the accuracy doesn't tends to increase much, which seems to be strange behaviour in this case since in the conventional deep learning based image classifier generally drop in loss bumps up the accuracy which doesn't seems to work in this case. Which seems to render the loss as intractable in this case.

Based on all the results I could determine one of the flaw in the method which is if the model size is more dense then during the pretraining the model might over fit and under fit which could lead to the subsequent performance drop in the domain adaptation setting. Hence, it could be deduced that the model is heavily dependent on the pre training part and since no method or term is used to regularize or compensate the the pre training of the models. Also, according to my speculation the training the models on all the domains might lead to more generalization than required forcing the CNNs to learn more robust features than required which leads to the poor performance and might require more time to train the models for that. Therefore if we could only apply pretraining on the source and target domain then in that case the fine tuning of the model could be done during the pre training leading a room to relatively small training steps for the PAC learning.

Important Links

- **Trained Weights**: It contains all the trained models on the PAC for using in the inference and validation.
- **Tensorboard Visualization**: This link contains all the training loss, accuracy for all the three models trained for rotation. It also contains more information about model architecture, histogram, kernel activation, etc.
- **Source Code**: It links to the GitHub repo containing source code of all the implemented code, which is updated in iterative fashion.

References

Samarth Mishra, Kate Saenko, and Venkatesh Saligrama. Surprisingly simple semi-supervised domain adaptation with pretraining and consistency, 2021.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.

Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8050–8058, 2019.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.

Yu Sun, Eric Tzeng, Trevor Darrell, and Alexei A Efros. Unsupervised domain adaptation through self-supervision. *arXiv preprint arXiv:1909.11825*, 2019.

Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. *Advances in neural information processing systems*, 29:1163–1171, 2016.

Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020.

Rui Shu, Hung H Bui, Hirokazu Narui, and Stefano Ermon. A dirt-t approach to unsupervised domain adaptation. *arXiv preprint arXiv:1802.08735*, 2018.

Vikas Verma, Kenji Kawaguchi, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. Interpolation consistency training for semi-supervised learning. *arXiv preprint arXiv:1903.03825*, 2019.

Jiaolong Xu, Liang Xiao, and Antonio M López. Self-supervised domain adaptation for computer vision tasks. *IEEE Access*, 7:156694–156706, 2019.

Bram Wallace and Bharath Hariharan. Extending and analyzing self-supervised learning across domains. In *European Conference on Computer Vision*, pages 717–734. Springer, 2020.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

Alex Clark. Pillow (pil fork) documentation, 2015. URL `https://buildmedia.readthedocs.org/media/pdf/pillow/latest/pillow.pdf`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016. doi: 10.1109/cvpr.2016.90. URL `http://dx.doi.org/10.1109/cvpr.2016.90`.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.